

[Claims]

What is claimed is:

5 1. A method for sharing execution capacity among tasks executing in a real-time computing system having a performance specification in accordance with Rate Monotonic Analysis (RMA), comprising the steps of:

10 pairing a higher priority task with a lower priority task;

reallocating execution time from the lower priority task to the higher priority task during an overload condition; and

15 increasing the period of the lower priority task to compensate for said reallocated execution time.

20 2. The method of claim 1, wherein an amount of said execution time available to loan from said lower priority task, task_R, to said higher priority task, task_u, is obtained as follows:

$$N_u = \frac{N_r \cdot T_u}{T_r}$$

where,

N_r = amount of execution time to borrow from task_r,

where N_r < C_r,

25 T_r = period of task_r, and

T_u = period of task_u.

30 3. The method of claim 1, wherein said increased period of the lower priority task, task_r, is obtained as follows:

$$T_n = \frac{C_r \cdot T_r}{C_r - N_r}$$

where

C_r = worst-case task execution time of task_r,

T_r = period of task_r, and

N_r = amount of execution time to borrow from task_r,

where $N_r < C_r$.

5

4. The method of claim 1, further comprising the step of limiting an amount of execution time, N_r , to borrow from said lower priority task, task_r, to a maximum loan amount where $N_r \ll C_r$, where

10

C_r = worst-case task execution time of task_r, and

N_r = amount of execution time to borrow from task_r.

15

5. The method of claim 4, wherein a maximum execution time, N_m , that may be borrowed from said lower priority task, task_r, is obtained as follows:

$$N_m = C_r \left(1 - \frac{1}{m} \right)$$

where m is the multiple of the period of said lower priority task, task_r.

20

6. The method of claim 1, wherein said higher priority task has hard deadlines.

25

7. The method of claim 1, wherein said lower priority task has soft deadlines.

30

8. A method for allocating resources among tasks executing in a real-time computing system having a performance specification in accordance with Rate Monotonic Analysis (RMA), comprising the steps of:

pairing a higher priority task with a lower priority task;

providing a first resource allocation to said lower priority task during a normal operating condition; and

reallocating a portion of said first resource allocation from said lower priority task to said higher priority task when said higher priority task is operable.

9. The method of claim 8, wherein said reallocated portion of said first resource allocation is obtained as follows:

$$N_u = \frac{N_r \cdot T_u}{T_r}$$

where,

N_r = amount of execution time to borrow from task_r,

where $N_r < C_r$,

T_r = period of task_r, and

T_u = period of task_u.

10. The method of claim 8, further comprising the step of increasing a period of said lower priority task, task_r, as follows:

$$T_n = \frac{C_r \cdot T_r}{C_r - N_r}$$

where

C_r = worst-case task execution time of task_r,

T_r = period of task_r, and

N_r = amount of execution time to borrow from task_r,

where $N_r < C_r$.

11. The method of claim 8, further comprising the step of limiting an amount of execution time, N_r , to reallocate from said lower priority task, task_r, to a maximum loan amount where $N_r \ll C_r$, where

C_r = worst-case task execution time of task_r, and

N_r = amount of execution time to borrow from task_r.

12. The method of claim 11, wherein a maximum execution time, Nm , that may be borrowed from said lower priority task, $task_r$, is obtained as follows:

$$5 \quad Nm = Cr \left(1 - \frac{1}{m} \right)$$

where m is the multiple of the period of said lower priority task, $task_r$.

13. The method of claim 8, wherein said higher priority
10 task has hard deadlines.

14. The method of claim 8, wherein said lower priority task
has soft deadlines.

15. A method for sharing execution capacity among tasks
executing in a real-time computing system having a performance
specification in accordance with Rate Monotonic Analysis (RMA),
comprising the steps of:

pairing a higher priority task, $task_u$, with a lower
20 priority task, $task_r$;

reallocating execution time from the lower priority
task to the higher priority task during an overload condition;
and

increasing the utilization of said higher priority
25 task; and

decreasing the utilization of said lower priority task
in a proportional manner to maintain a constant utilization, U .

16. The method of claim 15, wherein said utilizations of
30 said tasks are varied as follows:

$$\frac{Cu}{Tu} + \frac{Cr}{Tr} = U$$

where,

Sub
B1

C_u = worst-case task execution time of task_u,

T_u = period of task_u,

C_r = worst-case task execution time of task_r,

T_r = period of task_r, and

U = utilization for both tasks.

17. The method of claim 15, wherein an amount of said execution time available to reallocate from said lower priority task, task_r, to said higher priority task, task_u, is obtained as follows:

$$N_u = \frac{N_r T_u}{T_r}$$

where,

N_r = amount of execution time to borrow from task_r,

where $N_r < C_r$,

T_r = period of task_r, and

T_u = period of task_u.

18. The method of claim 15, further comprising the step of increasing a period of the lower priority task, task_r, as follows:

$$T_n = \frac{C_r \cdot T_r}{C_r - N_r}$$

where

C_r = worst-case task execution time of task_r,

T_r = period of task_r, and

N_r = amount of execution time to borrow from task_r,

where $N_r < C_r$.

19. The method of claim 15, further comprising the step of limiting an amount of execution time, N_r , to borrow from said

lower priority task, $task_r$, to a maximum loan amount where $N_r \ll C_r$, where

C_r = worst-case task execution time of $task_r$, and

N_r = amount of execution time to borrow from $task_r$.

5

20. The method of claim 19, wherein a maximum execution time, N_m , that may be borrowed from said lower priority task, $task_r$, is obtained as follows:

$$N_m = C_r \left(1 - \frac{1}{m} \right)$$

10 where m is the multiple of the period of said lower priority task, $task_r$.

21. The method of claim 15, wherein said higher priority task has hard deadlines.

22. The method of claim 15, wherein said lower priority task has soft deadlines.

23. A real-time computing system having a performance specification in accordance with Rate Monotonic Analysis (RMA), comprising:

a memory for storing computer readable code; and

a processor operatively coupled to said memory, said processor configured to:

25 pair a higher priority task with a lower priority task; reallocate execution time from the lower priority task to the higher priority task during an overload condition; and

increase the period of the lower priority task to compensate for said reallocated execution time.

30

24. A real-time computing system having a performance specification in accordance with Rate Monotonic Analysis (RMA), comprising:

a memory for storing computer readable code; and
 5 a processor operatively coupled to said memory, said processor configured to:

pair a higher priority task with a lower priority task;
 provide a first resource allocation to said lower
 priority task during a normal operating condition; and

10 reallocate a portion of said first resource allocation from said lower priority task to said higher priority task when said higher priority task is operable.

25. A real-time computing system having a performance specification in accordance with Rate Monotonic Analysis (RMA), comprising:

a memory for storing computer readable code; and
 a processor operatively coupled to said memory, said
 processor configured to:

20 pair a higher priority task, task_u , with a lower priority task, task_r ;

reallocate execution time from the lower priority task to the higher priority task during an overload condition; and

increase the utilization of said higher priority task;

25 and

decrease the utilization of said lower priority task in a proportional manner to maintain a constant utilization, U .

00110-122150